```
=> d his full


        (FILE 'USPAT' ENTERED AT 10:07:41 ON 05 JUN 1998)
                    DELETE HISTORY Y
L1            13 SEA (4994963 OR 5708780 OR 5721908 OR 5710918 OR 5276879 O
R 5

              734865 OR 5701451 OR 5636371 OR 5471615 OR 5727129 OR 5737
560

                 OR 5666530 OR 5280627)/PN
L2            13 SEA (5146568 OR 5734871 OR 5706502 OR 5655081 OR 5228137 O
R 5

              673322 OR 4780821 OR 5734835 OR 5230052 OR 5161116 OR 5276
863

                 OR 5572572 OR 5511202)/PN
L3            13 SEA (5386360 OR 5400246 OR 5361359 OR 5410706 OR 5577220 O
R 5

              544320 OR 5513359 OR 5727159 OR 5689708 OR 4949248 OR 5497
494

                 OR 5640537 OR 5696901)/PN
L4            11 SEA (5724506 OR 5522075 OR 5195130 OR 5073933 OR 5548763 O
R 5

              675510 OR 5247616 OR 4991089 OR 5732212 OR RE35448 OR 5732
219

                 )/PN
L5            50 SEA L1 OR L2 OR L3 OR L4
L6           342 SEA (395/186,187.01,188.01)/CCLS
L7          2694 SEA (395/200.?)/CCLS
L8          1783 SEA (380/3,4,23,25)/CCLS
L9            97 SEA HOST? (P)CLIENT# SERVER#
L10            1 SEA L9 AND JAVA
L11         4613 SEA L6 OR L7 OR L8
L12           10 SEA L11 AND JAVA
L13            9 SEA L12 NOT L5
```

1. 5,761,421, Jun. 2, 1998, System and method for secure peer-to-peer communication between downloaded programs; Arthur A. van Hoff, et al., **395/200.53**; 340/825.5; **395/200.58** [IMAGE AVAILABLE]

US PAT NO: 5,761,421 [IMAGE AVAILABLE]   L12: 1 of 10

ABSTRACT:
A system and method for establishing a peer-to-peer communication connection between computer programs from the same security domain, but executing in first and second computers, is disclosed. A first computer program, while executing in the first computer, sends a communication a message to the second computer, requesting a peer-to-peer communication connection. Upon receiving the message at said second computer, the second computer determines whether a second computer program meeting predefined criteria for establishing a peer-to-peer communication connection is executing in the second computer. If so, the second computer sends to the first computer a reply message accepting the request. After receipt of the reply message by the first computer, the requested peer-to-peer communication connection between the first and second computer programs is established. In a preferred embodiment, the predefined criteria for establishing a peer-to-peer communication connection is that the first and second computer programs be from the same server computer.

2. 5,745,703, Apr. 28, 1998, Transmission of higher-order objects across a network of heterogeneous machines; Henry Cejtin, et al., **395/200.68, 200.31, 200.43, 200.75**; 707/10, 201, 202 [IMAGE AVAILABLE]

US PAT NO: 5,745,703 [IMAGE AVAILABLE]   L12: 2 of 10

ABSTRACT:
The system comprises a collection of address spaces within which potentially many concurrent lightweight perceptible threads may execute. The address space is uniformly distributed among different nodes in a network of heterogeneous machines. Address spaces are first-class and may be generated dynamically. Threads within an address space may communicate with one another via shared memory; communication between address spaces takes place using explicit message-passing.

3. 5,745,569, Apr. 28, 1998, Method for stega-cipher protection of computer code; Scott A. Moskowitz, et al., **380/4, 23, 25,** 28, 49, 50, 54 [IMAGE AVAILABLE]

US PAT NO: 5,745,569 [IMAGE AVAILABLE]   L12: 3 of 10

ABSTRACT:
A method for protecting computer code copyrights by encoding the code into a data resource with a digital watermark. The digital watermark contains licensing information interwoven with essential code resources encoded into data resources. The result is that while an application program can be copied in an uninhibited manner, only the licensed user having the license code can access essential code resources to operate the program and any descendant copies bear the required license code.

4. 5,740,248, Apr. 14, 1998, Software level touchpoints for an international cryptography frameworks; Helmut Fieres, et al., **380/25,**

US PAT NO: 5,740,248 [IMAGE AVAILABLE]

ABSTRACT:
An international cryptography framework (ICF) allows manufacturers to
comply with varying national laws governing the distribution of
cryptographic capabilities. The invention is concerned primarily with the
application certification aspects of the framework where an application
that requests cryptographic services from the ICF service elements is
identified through some form of certificate to protect against the misuse
of a granted level of cryptography. The levels of cryptography granted
are described via security policies and expressed as classes of service.
A cryptographic unit, one of the ICF core elements, can be used to build
several certification schemes for application objects. The invention
provides various methods that determine the strength of binding between
an application code image and the issued certificates within the context
of the ICF elements. A key element with regard to the exercise of a
cryptographic function concerns the special requirements for the trust
relation that an authority specifies for the cryptographic unit. Any
function exercised by the cryptographic unit must be controllable by the
associated class of service which represents the security policy.
Touchpointing, both in the application and the firmware elements inside
the cryptographic unit, plays a key role in exercising control over the
functioning of these modules. Another fundamental requirement of the ICF
architecture is that the application is assured of the integrity of the
cryptographic unit from which it is receiving services. Thus, the
invention also provides methods that allow a determination of whether or
not the cryptographic unit has been replaced or tampered with.

5. 5,734,835, Mar. 31, 1998, Inexpensive world wide web terminal
appliance using disk synchronized with raster refresh for storage of
displayed pages; Edwin Joseph Selker, **395/200.79**; 345/2, 213;
348/464, 476, 569; 360/73.03; **395/200.47, 200.48, 200.49**;
711/112 [IMAGE AVAILABLE]

US PAT NO: 5,734,835 [IMAGE AVAILABLE]

ABSTRACT:
A World Wide Web terminal appliance utilizes a disk drive for local
storage of Web pages previously downloaded and rendered for display
during the course of a Web surfing session. The disk drive rotates at a
rate substantially in synchronization with a display refresh time
interval of a display device, preferably a raster refresh cycle time of a
video monitor. Therefore, the image being displayed need not occupy
random-access memory, but rather is sent directly from the disk to a
display interface for coupling to the display device. Little or no RAM
buffering is required, so the appliance need not include a large quantity
of video RAM storage. A relatively inexpensive disk is used instead,
thereby bringing about advantageous cost savings.

6. 5,727,147, Mar. 10, 1998, System and method for resolving symbolic
references to externally located program files; Arthur A. van Hoff,
395/200.3, 200.42, 200.47 [IMAGE AVAILABLE]

US PAT NO: 5,727,147 [IMAGE AVAILABLE]

ABSTRACT:
When an interpreter on a client computer encounters a symbolic reference
to a remotely stored method while interpreting a locally stored method,
and the object class for the remotely stored method has not previously
been loaded, the client computer, the client computer creates an
application specific loader that is then used to load the remotely stored
method into the client computer. The application specific class loader
contains location information associated with the server computer on

which the remotely stored method is stored, and also contains methods for
loading onto the client computer the object class for the remotely stored
method as well as the object classes for any additional methods
referenced by that method. The application specific class loader
preferably also includes symbol table for storing information about
method references that have been resolved by the application specific
class loader. Each method in the object classes loaded by the application
specific class loader is modified when it is loaded to reference the
application specific class loader, thereby linking the loaded method to
the application specific class loader. The reference to the application
specific class loader in the loaded method is used by the interpreter to
determine which class loader to use when resolving symbolic references in
the loaded method to other methods. The methods of the application
specific class loader may incorporate load policies associated with the
server location and different from the load policies used by the client
computer's bootstrap class loader.

7.   5,708,709, Jan. 13, 1998, System and method for managing try-and-buy
usage of application programs; John R. Rose, **380/4**, 9, **23,
25,** 30, 49 [IMAGE AVAILABLE]

US PAT NO: 5,708,709 [IMAGE AVAILABLE]   L12: 7 of 10

ABSTRACT:
A system and method for managing the distribution of licensed application
programs stored on a server over a distributed computer system maintains
control over the program even after the program has been distributed to a
client computer from a provider on an information server. Protection may
include license expiration date verification, authorized user ID
verification, and protection against decompilation and reverse
engineering by maintaining the program in an encrypted form until
verification of the expiration date and user identity are complete and
the program is ready for decoding, loading into the client computer CPU,
and execution. A user identifies a program for trial use by any
conventional means such as by using a network browser on the World Wide
Web. The server recognizes a user request to access the application
program. The server may have an agent on the client computer for
performing certain predetermined administrative tasks. This agent may
take the form of an application builder program module, provided by the
trial application provider, which is resident on the client computer. The
server (including the agent) determines whether program access conditions
are satisfied, and if satisfied transmits a version of the program to the
client. The transmitted file includes an encrypted portion. The server
and agent also verify that the user is currently entitled to execute the
application program including that the trial license has not expired at
the time the user initiates execution, and generates an executable
version of the application program.

8.   5,706,434, Jan. 6, 1998, Integrated request-response system and
method generating responses to request objects formatted according to
various communication protocols; Gary Kremen, et al., **395/200.48,
200.36, 200.6, 200.76,** 285, 500; 707/10 [IMAGE AVAILABLE]

US PAT NO: 5,706,434 [IMAGE AVAILABLE]   L12: 8 of 10

ABSTRACT:
A method and apparatus is provided to accomplish creation and serving of
data objects among various communication protocols. The method and
apparatus can be used in such applications as an on-line classified
advertising system on the Internet involving the World Wide Web and
electronic mail. In the apparatus, a request decoder receives an incoming
request, decodes the request using configurations from a configuration
database in order to identify which protocol was used to transmit the
request, and generates from the request a corresponding abstract data
object. A data processor merges data from a main database with the

8. 5,706,434, Jan. 6, 1998, Integrated request-response system and
method generating responses to request objects formatted according to
various communication protocols; Gary Kremen, et al., **395/200.48,
200.36, 200.6, 200.76,** 285, 500; 707/10 [IMAGE AVAILABLE]

US PAT NO: 5,706,434 [IMAGE AVAILABLE]   L12: 8 of 10

ABSTRACT:
A method and apparatus is provided to accomplish creation and serving of
data objects among various communication protocols. The method and
apparatus can be used in such applications as an on-line classified
advertising system on the Internet involving the World Wide Web and
electronic mail. In the apparatus, a request decoder receives an incoming
request, decodes the request using configurations from a configuration
database in order to identify which protocol was used to transmit the
request, and generates from the request a corresponding abstract data
object. A data processor merges data from a main database with the
abstract data object. An object formatter formats the abstract data
object including the merged data. An object deliverer formats the object
for outgoing transmission according to a protocol of an intended
recipient. The functions of object deliverer may be performed by the
object formatter.

9. 5,692,047, Nov. 25, 1997, System and method for executing verifiable
programs with facility for using non-verifiable programs from trusted
sources; Charles E. McManis, **380/4; 395/186,** 704 [IMAGE
AVAILABLE]

US PAT NO: 5,692,047 [IMAGE AVAILABLE]   L12: 9 of 10

ABSTRACT:
A computer system includes a program executer that executes verifiable
architecture neutral programs and a class loader that prohibits the
loading and execution of non-verifiable programs unless (A) the
non-verifiable program resides in a trusted repository of such programs,
or (B) the non-verifiable program is indirectly verifiable by way of a
digital signature on the non-verifiable program that proves the program
was produced by a trusted source. In the preferred embodiment, verifiable
architecture neutral programs are **Java** bytecode programs whose
integrity is verified using a **Java** bytecode program verifier. The
non-verifiable programs are generally architecture specific compiled
programs generated with the assistance of a compiler. Each architecture
specific program typically includes two signatures, including one by the
compiling party and one by the compiler. Each digital signature includes
a signing party identifier and an encrypted message. The encrypted
message includes a message generated by a predefined procedure, and is
encrypted using a private encryption key associated with the signing
party. A digital signature verifier used by the class loader includes
logic for processing each digital signature by obtaining a public key
associated with the signing party, decrypting the encrypted message of
the digital signature with that public key so as generate a decrypted
message, generating a test message by executing the predefined procedure
on the architecture specific program associated with the digital
signature, comparing the test message with the decrypted message, and
issuing a failure signal if the decrypted message digest and test message
digest do not match.

10. 5,680,461, Oct. 21, 1997, Secure network protocol system and method;

US PAT NO: 5,680,461 [IMAGE AVAILABLE]   L12: 10 of 10

ABSTRACT:
A computer network having first and second network entities. The first
network entity includes a packet object generator that generates a packet
object including an executable source method, an executable destination
method, and data associated with each of the methods. The first network
entity also includes a communications interface to transmit the packet
object. The second network entity includes a communications interface to
receive the packet object and an incoming packet object handler to handle
the received packet object. The incoming packet object handler includes a
source and destination verifier to execute the source and destination
methods with their associated data so as to verify the source and
destination of the received object packet.

=> d his full

=> d cit,fd,ab,kwic

1. 5,754,830, May 19, 1998, Server and web browser terminal emulator for persistent connection to a legacy host system and method of operation; Thomas H. Butts, et al., 395/500; 370/466, 469; 395/285, 680 [IMAGE AVAILABLE]

US PAT NO: 5,754,830 [IMAGE AVAILABLE]   L10: 1 of 1
DATE FILED: Apr. 1, 1996

ABSTRACT:
A computer network environment (10) allowing connection of a client system (36) to a legacy host system (18,19) using a server (26) is provided. The computer network environment (10) includes a legacy host system (18,19) having TCP/IP connectivity. The legacy host system (18,19) is operable to support a terminal session for access to the legacy host system (18,19). The computer network environment (10) also includes a server system (24) executing a client thread (28) under a server (26). The client thread (28) is operable to communicate with the legacy host system (18,19) across a persistent TCP/IP socket connection (30). The computer network environment (10) further includes a client system (36) executing an applet process (42) under a web browser (38). The applet process (42) is operable to communicate with the client thread (28) across another persistent TCP/IP socket connection (44) and is operable to provide a terminal session to a user of the client system (36). This terminal session is supported by a persistent connection allowing real-time bidirectional communication with the legacy host system (18,19).

SUMMARY:

BSUM(12)

 A . . . security, encryption, help-desk support, and other real-time features can be supported. One embodiment of the present invention uses SUN MICROSYSTEMS' **JAVA** technology and includes **JAVA**-capable web browsers and embedded **JAVA** applet processes to provide terminal session connectivity to the distributed client systems.

SUMMARY:

BSUM(14)

 A . . . invention provides a network environment that allows the use of a web browser environment, having web browser tools, such as **JAVA** tools, and web-serving, to incorporate Internet-type technologies, through the Internet or an intranet, with existing network architectures.

DETDESC:

DETD(6)

 Internet/intranet . . . process. Web/emulation server 26 can comprise an OC://WEBCONNECT.TM. server available from OPENCONNECT SYSTEMS, and applet code 34 can comprise a **JAVA** applet for use within SUN MICROSYSTEM's **JAVA** environment.

DETDESC:

DETD(7)

Public . . . socket connection 44. Web browser 38 can comprise a
commercially available web browser application such as NETSCAPE NAVIGATOR
that is **JAVA**-capable and applet process 42 can comprise a **JAVA**
applet.

DETDESC:

DETD(11)

According . . . updates, security, encryption, help-desk support, and
other real-time features are supported. One embodiment of the present
invention uses SUN MICROSYSTEMS' **JAVA** technology and includes
JAVA-capable web browsers 38 and embedded **JAVA** applet processes
42 to provide terminal session connectivity to client systems 36.

DETDESC:

DETD(13)

The . . . a network environment 10 that allows the use of a public
Internet/intranet environment 16 having web browser tools, such as
JAVA tools, and web-serving to incorporate Internet-type
technologies, through the Internet or an intranet, with existing network
architectures. Thus, an organization. . .

DETDESC:

DETD(15)

In . . . legacy host system is selected from a web browser executing
on a client system. The web browser can comprise a **JAVA**-capable
NETSCAPE NAVIGATOR web browser as mentioned above. The selected uniform
resource locator is received by a web/emulator server in step. . . In
step 56, the client system executes the applet process under the web
browser. The applet process can comprise a **JAVA** applet for execution
within a **JAVA** virtual machine within the NETSCAPE NAVIGATOR web
browser.

DETDESC:

DETD(20)

The . . . data flows into a terminal session for display to the user.
As mentioned above, the web browser can comprise a **JAVA**-capable web
browser, the applet process can be a **JAVA** applet, and the
web/emulator data flow can be based upon the protocol set forth in
Appendix A. This embodiment of the present invention can blend web
browsers enhanced by SUN MICROSYSTEMS' **JAVA** with legacy host systems
having TCP/IP connectivity to allow users on any client system platform
to connect to and access. . .

DETDESC:

DETD(23)

The . . . time is improved by taking advantage of an existing
widely-installed web browser base. Specifically, the platform
independence provided by the **JAVA** architecture allows the web browser
terminal emulator of the present invention to avoid problems with
operation differences between computer environments.. . .

DETDESC:

DETD(24)

The present invention provides connectivity from any client system, such as a personal computer or computer workstation, to a legacy **host** system, such as a mainframe or mid-range system, without costly redesign or rebuilding of legacy applications. The benefits of a distributed **client/server** type data exchange can be realized without requiring systems redesign. This approach means that the many advantages of Internet-type access. . .

CLAIMS:

CLMS(4)

4. The server of claim 3, wherein the applet code comprises executable code for a **JAVA** applet to be executed under a **JAVA**-capable web browser.

CLAIMS:

CLMS(12)

12. The web browser terminal emulator of claim 11, wherein the applet process comprises a **JAVA** applet executing under a NETSCAPE NAVIGATOR web browser.

CLAIMS:

CLMS(17)

17. The computer network environment of claim 16, wherein the applet process comprises a **JAVA** applet executing under a NETSCAPE NAVIGATOR web browser.

CLAIMS:

CLMS(28)

28. The method of claim 27, wherein executing the applet process comprises executing a **JAVA** applet under a NETSCAPE NAVIGATOR web browser.